OpenSource Implementation of MS RIS Server

Gianluigi Tiesi

May 29, 2007

Contents

1 Preface

This guide explains the Remote Installation of all supported Windows versions: Windows 2000, Windows XP and Windows 2003.
If you only need to setup one Operating System, just skip references to other ones.
The setup is tested on Gnu/Debian Linux (but it should work fine also with other distributions) and on Windows XP (Every platform supported by tftpd32 should work fine too).
Since the code is endian independent you should be able to make it work with most unix-like systems.
On Windows platform you may need to install Python, I suggest you to use ActiveState Python since it comes with a nice setup and most of win32 extensions.
You may also need some tools for unix like sed or cabextract to extract cab compressed files, you can also use an archive unpacker extracting files, I just recommend 7zip, since it is free and has a lot of nice features.
You will find references to a server name "attila", so you should replace it with your own server name/netbios name. You will also find reference to my own network 192.168.129.0/24, so just replace as needed.

2 Requirements

- My toolset of utility is called ris-linux, the package can be found on my web page: http://oss.netfarm.it/.
- Linux/Unix box (I suggest Gnu/Debian) or a Windows win2k+ System, I really do not known if it works on win9x
- Python version $\geq$ 2.1 but you can still use a minimal service written in C
- A dhcp server: ISC v3 or DNSMasq
- A tftp server: atftp or hpa tftp on unix, tftpd32 on Windows
- A working Samba server, Windows is able to share by itself
- PXELinux very useful to make multiple choices menus

3 Setup

3.1 Samba / Windows Shares

3.1.1 Samba Configuration

```
[REMINST]
     path = /mnt/disk/ris
     browsable = true
     read only = No
     guest ok = Yes
```

You must set null **passwords = true** in the global section or the ris client will not able to access files. **read only** option is not really needed but this way you should be able to copy setup files directly from a Windows box.

3.1.2 Windows Shares

Make a directory for tftp root, use tftpboot as name, then share it as RemInst. Please note that XP SP2 (and maybe other future windows versions/service packs) needs a special registry patch to re-enable "null session" shares, that is required to make RIS work. To achieve this you need to merge this registry file (copy and paste in a notepad, then save it with .reg extension). A reboot is required in order to make these changes have effect. The reg file is also in my software package.

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters]
"NullSessionPipes"=hex(7):43,00,4f,00,4d,00,4e,00,41,00,50,00,00,00,43,00,4f,\
  00,4d,00,4e,00,4f,00,44,00,45,00,00,00,53,00,51,00,4c,00,5c,00,51,00,55,00,\
  45,00,52,00,59,00,00,00,53,00,50,00,4f,00,4f,00,4c,00,53,00,53,00,00,00,4c,\
  00,4c,00,53,00,52,00,50,00,43,00,00,00,62,00,72,00,6f,00,77,00,73,00,65,00,\
  72,00,00,00,73,00,72,00,76,00,73,00,76,00,63,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa]
"RestrictAnonymous"=dword:00000000
```

3.2 TFTP Server

The following procedures apply to a debian sid, but you can adapt it to whatever OS you want, just skip debian specific operations: download sources in place of apt-get source and

`./configure && make && make install in place of dpkg -i.`

You first need to download package source.

3.2.1 atftpd

▷ http://packages.debian.org/unstable/net/atftpd

```
apt-get update
...
apt-get source atftpd
gpg: Signature made Mon 12 Feb 2007 10:13:47 AM CET using DSA key ID C0143D2D
gpg: Can't check signature: public key not found
dpkg-source: extracting atftp in atftp-0.7.dfsg
dpkg-source: unpacking atftp_0.7.dfsg.orig.tar.gz
dpkg-source: applying ./atftp_0.7.dfsg-1.2.diff.gz

apt-get build-dep atftpd
...
cd atftp-0.7.dfsg
patch -p1 < ../ris-linux/atftpd-filecase.diff

Hunk #1 succeeded at 659 (offset 4 lines).
patching file tftpd.h
patching file tftpd_file.c

dpkg-buildpackage -b

dpkg-deb: building package 'atftp' in '../atftp_0.7.dfsg-1.2_i386.deb'.
dpkg-deb: building package 'atftpd' in '../atftpd_0.7.dfsg-1.2_i386.deb'.

dpkg -i ../atftpd_0.7.dfsg-1.2_i386.deb
```

3.2.2 hpa tftp

▷ http://www.kernel.org/pub/software/network/tftp/
Please note that hpa tftpd does not allow soft links in the directory tree.

```
apt-get update
...
apt-get source tftp-hpa
gpg: Signature made Sat 24 Feb 2007 11:10:22 AM CET using DSA key ID C0143D2D
gpg: Can't check signature: public key not found
dpkg-source: extracting tftp-hpa in tftp-hpa-0.43
dpkg-source: unpacking tftp-hpa_0.43.orig.tar.gz
dpkg-source: applying ./tftp-hpa_0.43-1.1.diff.gz

apt-get build-dep tftp-hpa
...
cd tftp-hpa-0.43
patch -p1 < ../ris-linux/tftp-hpa-filecase.diff

patching file tftpd/tftpd.c
Hunk #1 succeeded at 61 with fuzz 2.
Hunk #2 succeeded at 776 (offset 62 lines).
Hunk #3 succeeded at 1115 (offset 62 lines).
Hunk #4 succeeded at 1199 (offset 62 lines).

dpkg-buildpackage -b

pkg-deb: building package 'tftp-hpa' in '../tftp-hpa_0.43-1.1_i386.deb'.
dpkg-deb: building package 'tftpd-hpa' in '../tftpd-hpa_0.43-1.1_i386.deb'.

dpkg -i ../tftpd-hpa_0.43-1.1_i386.deb
```

3.2.3 Windows tftpd32

▷ http://tftpd32.jounin.net/
Unpack the tftpd32 archive to a directory, click on “Settings”, then change options as described in Figure 1, select the directory previously shared as Base Directory.

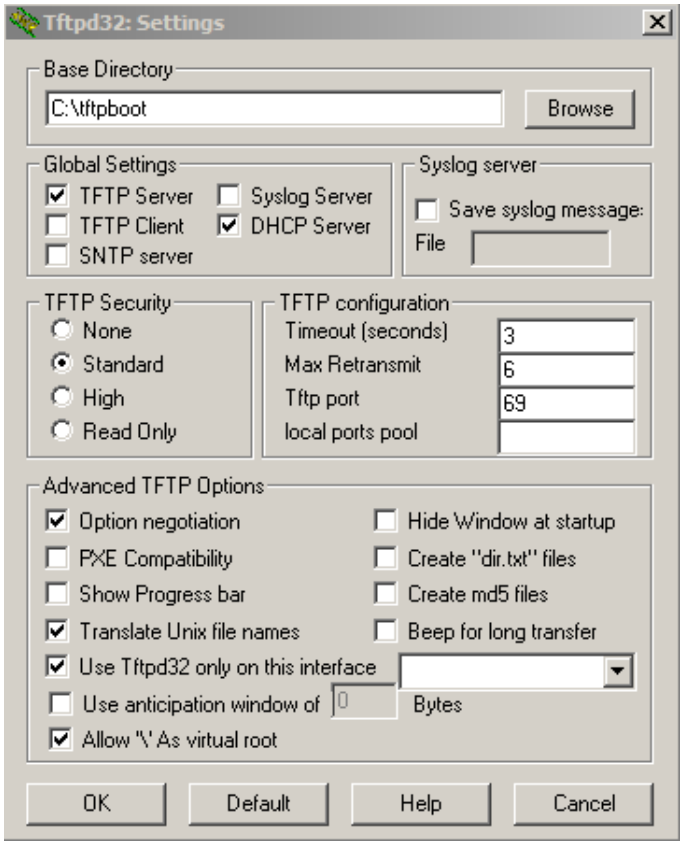

Figure 1: tftpd32 settings

3.3 DHCP Server

3.3.1 ISC DHCP v3

▷ http://www.isc.org/index.pl?/sw/dhcp/

```
subnet 192.168.129.0 netmask 255.255.255.0
{
  range dynamic-bootp 192.168.129.100 192.168.129.110;
  default-lease-time 600;
  max-lease-time 7200;
  filename "pxelinux.0";
  server-name "attila.of.war";
  next-server attila.of.war;
}
```

3.3.2 DnsMasq

▷ http://thekelleys.org.uk/dnsmasq/doc.html

```
dhcp-range=192.168.129.100,192.168.129.150,1h
dhcp-boot=pxelinux.0,attila,192.168.129.1
```

3.3.3 Windows tftpd32 DHCP

▷ http://tftpd32.jounin.net/
Click on DCHP server tab, then fill values as described in Figure 2, change you network interface, starting pool, size, mask as needed. Use pxelinux.0 as Boot File. Please note that if you have enabled ICS (Internet Connection Sharing), tftpd32 will not work because ICS does dhcp server too.

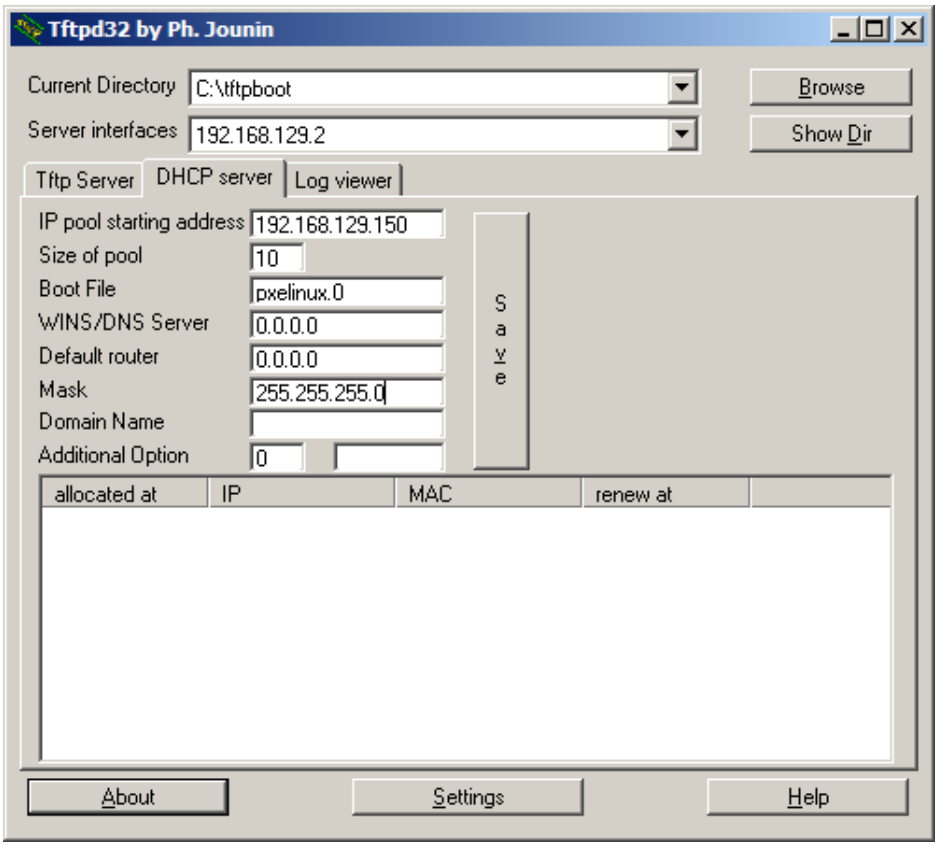

Figure 2: tftpd32 dchp

4 Stage

4.1 Needed files

- pxelinux.0 and pxelinux.cfg/default You can find pxelinux.0 in the syslinux package, http://syslinux.zytor.com/. pxelinux.cfg/default should be like:

```
label win2k
    kernel w2k.0
label winxp
    kernel winxp.0
label win2k3
    kernel w2k3.0
```

- w2k.0, wxp.0, w23.0 Locate a file called *i386/STARTROM.N1_* in the Windows installation directory/ Use cabextract as follow or you favourite unpacker:

  ```
  cabextract <Source dir>/i386/STARTROM.N1\_
  ```

  so you will have a file called *startrom.n12*.
  Change the ntloader filename to be able to use multiple setups, you will need sed command line utility: http://gnuwin32.sourceforge.net/packages/sed.htm.
  Windows 2000:

  ```
  sed -i -e 's/NTLDR/W2KLD/gi' startrom.n12
  ```

  Windows XP:

  ```
  sed -i -e 's/NTLDR/XPLDR/gi' startrom.n12
  ```

  Windows 2003:

  ```
  sed -i -e 's/NTLDR/W2K3L/gi' startrom.n12
  ```

  Then rename startrom.n12 to w2k.0, wxp.0 or w23.0 depending on the Windows source installation you used.

- W2KLD, XPLDR, W2K3L Locate a file called *i386/SETUPLDR.EX_* in the Windows installation directory/ Use cabextract as follow or you favourite unpacker:

  ```
  cabextract <Source dir>/i386/SETUPLDR.EX\_
  ```

  so you will have a file called *setuploader.exe*.
  Change the response filename to be able to use multiple setups:
  Windows XP:

  ```
  sed -i -e 's/winnt\.sif/winxp\.sif/gi' setupldr.exe
  ```

  Windows 2003:

  ```
  sed -i -e 's/winnt\.sif/wi2k3\.sif/gi' setupldr.exe
  ```

  Then rename setuploader.exe to XPLDR or W2K3L depending on the Windows source installation you used. A special case is Windows 2000, there is bug(?) in win2k setup loader that appends the needed filename, to the string *pxelinux.0*, so just rename win2k setuploader to W2KLD for now, you will need to use different names for the ntdetect and the response file.

- pxelinux.0ntdetect.com ntdetect.wxp ntdetect.2k3

  Please note that win2k needs ntdetect.com file named as *pxelinux.0ntdetect.com*. *ntdetect.com* is located in *¡Source dir¿/i386/*. You should change the response filename, as before. No changes for Windows 2000.
  Windows XP:

  ```
  sed -i -e 's/ntdetect\.com/ntdetect\.wxp/gi' setupldr.exe
  ```

  Windows 2003:

  ```
  sed -i -e 's/ntdetect\.com/ntdetect\.2k3/gi' setupldr.exe
  ```

- pxelinux.0winnt.sif winxp.sif wi2k3.sif

  Use this for all response files:

```
[data]
floppyless = "1"
msdosinitiated = "1"
; Needed for second stage
OriSrc = "\\SERVERNAME\SHARENAME\<setup_dir>\i386"
OriTyp = "4"
LocalSourceOnCD = 1
DisableAdminAccountOnDomainJoin = 1

[SetupData]
OsLoadOptions = "/fastdetect"
; Needed for first stage
SetupSourceDevice = "\Device\LanmanRedirector\SERVERNAME\SHARENAME\<setup_dir>"

[UserData]
ComputerName = *
; if needed
;ProductID=
```

Then replace ⟨*setup_dir*⟩, with win2k, winxp, win2k3 in each response file. Also replace *SERVERNAME* and *SHARENAME* with your own.

4.2 tftp directory tree

This should be the tftp root directory tree. Please note that files on the top level must be have the same case on unix since my case insensitive lookup patch only applies when a path starts by \.

▷tftproot/	
pxelinux.0	
▷ pxelinux.cfg/	
default	
▷ win2k/	
i386	← Windows 2000 Installation Files
▷ winxp/	
i386	← Windows XP Installation Files
▷ win2k3/	
i386	← Windows 2003 Installation Files
w2k.0	← startrom.n12 from Windows 2000
wxp.0	← startrom.n12 from Windows XP
w23.0	← startrom.n12 from Windows 2003
W2KLD	← setup loader from Windows 2000
XPLDR	← setup loader from Windows XP
W2K3L	← setup loader from Windows 2003
pxelinux.0ntdetect.com	← Windows 2000 ntdetect.com
ntdetect.wxp	← Windows XP ntdetect.com
ntdetect.2k3	← Windows 2003 ntdetect.com
pxelinux.0winnt.sif	← Windows 2000 Response File
winxp.sif	← Windows XP Response File
wi2k3.sif	← Windows 2003 Response File

Table 1: directory tree

4.3 Drivers

To make network install working you first need to collect needed inf files from windows inf directory or from downloaded networks drivers. After you placed needed inf files in a directory, you should run *infparser.py* to make the driver cache needed by the binl server. Same utility can be used to generate a txt file that can be used with the mini binl server.
Example, if you have placed your inf files in */mnt/disk/ris/inf* you should run:

```
infparser.py /mnt/disk/ris/inf
```

Then you will have two files: *devlist.cache* needed by the binlsrv (Python version) and *nics.txt* for the native mini binl server.
You need to place your binary *.sys* driver files in *i386* directory of the selected install path e.g. in */mnt/disk/ris/winxp/I386*

5 Remote Install

5.1 BINL Server

Microsoft BINL protocol handles a lot of request/response, but we only need one of them. The client detects the network card by scanning pci bus, the it ask binl server for a driver that matches Vendor ID and Product ID. So we can launch our binl server replacement as follows:

```
./binlsrv.py
Succesfully loaded 662 devices
Binlserver started...
```

You are now ready to boot your client from network, when pxelinux.0 will show the prompt select the label corresponding the operating system you want to install. If your motherboard/network card does not support booting from network you can make a floppy that emulates network boot using Rom-O-Matic http://rom-o-matic.net/. The client will start to ask for some files to the tftpd server, then you will see something in the binlsrv output:

```
Recv NCQ len = 48
NCQ Driver request
[R] Vid: 0x1022
[R] Pid: 0x2000
[R] rev_u1 = 0x2
[R] rev_u2 = 0x0
[R] rev_u3 = 0x0
[R] rev    = 0x10
[R] rev2   = 0x88
[R] subsys = 0x20001022
Checking PCI\VEN_1022&DEV_2000&SUB_20001022
Checking PCI\VEN_1022&DEV_2000
Found PCI\VEN_1022&DEV_2000 in netamd2.inf
[S] Packet len = 0xc4 (196)
[S] Result code: 0x0
[S] type: 0x2
[S] base offset = 0x24 (36)
[S] drv_off = 0x50 (80)
[S] srv_off: 0x6a (106) -> 98 from start
[S] plen: 0x56 (86)
[S] p_off: 0x76 (118) -> 110 from start
[S] hid: PCI\VEN_1022&DEV_2000 - Len 0x15 (21)
[S] drv: pcntpci5.sys - Len 0xc (12)
[S] srv: PCnet - Len 0x5 (5)
[S] Len data now -2 = 0x54 (84)
[S] Description (REG_EXPAND_SZ [2]) = Scheda Ethernet PCI AMD PCNET Family
[S] Characteristics (REG_SZ [1]) = 132
[S] BusType (REG_SZ [1]) = 5
[S] Total Params: 3
```

As you can see the client asked for a network card that has 1022 as Vendor ID and 2002 as Product ID. Vendor ID 1022 is AMD while 2000 means pcnet family. Your setup should and you will able to continue Windows installation.

6 Advanced

6.1 Client Modding

Windows installer client normally uses UDP port 4011 to talk with BINL server. In your network setup, you may need to use different port. You can also avoid to use sed to modify setuploader name and response file.

```
./modldr.py: [-l loader] [-p port] [-v] [-r response] inputfile
startrom:
 - changes SetupLoader       [-l] : exactly 5 chars - only if not yet modified
SetupLDR:
 - changes the udp port      [-p] : 16 bit integer
 - displays the current port [-v] : no modification are made
 - change response file      [-r] : exactly 9 chars - only if not yet modified

!!! Warning modifications are made in-place, so the file is modified !!!
```

SetupLoader is NTLDR or whatever you renamed it. If you use a different port in the client you should use same port on binlsrv.py, by adding -p port command line option. You can also use -a to bind BINL server to a specific interface. This way you can run multiple instances of BINL server listening to different port/interfaces.

6.2 Advanced BINL Server

```
Usage ./binlsrv.py: [-h] [-d] [-l logfile] [-a address] [-p port]
                    [--pid pidfile] [devlist.cache]
-h, --help     : show this help
-d, --daemon   : daemonize, unix only [false]
-l, --logfile= : logfile when used in daemon mode [/var/log/binlsrv.log]
-a, --address= : ip address to bind to [all interfaces]
-p, --port=    : port to bind to [4011]
    --pid=     : pid file to use instead of the default
devlist.cache  : device list cache file [devlist.cache in current dir]
```

As explained before you can run different instances on different addresses and ports, and you can specify different devlist cache files, this is usefull for example if network card drivers are incompatibles between differents OS to install (WinXP and Win2k3).